



AU 00 | 651

4

REC'D 30 JUN 2000

WIPO

PCT

Patent Office
Canberra

I, ANNA MAIJA EVERETT, ACTING TEAM LEADER EXAMINATION SUPPORT & SALES hereby certify that annexed is a true copy of the Provisional specification in connection with Application No. PQ 0917 for a patent by BRADFORD CRAIG STARKIE filed on 11 June 1999.



WITNESS my hand this
Twenty-sixth day of June 2000

ANNA MAIJA EVERETT
ACTING TEAM LEADER
EXAMINATION SUPPORT & SALES

**PRIORITY
DOCUMENT**

SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH RULE 17.1(a) OR (b)

AUSTRALIA
Patents Act 1990

COMPLETE SPECIFICATION
STANDARD PATENT

Development Environment for Natural
Language Dialogue Systems.

The invention is described in the following statement.

Development Environment for Natural Language Dialogue Systems.

1 This invention relates to software tools used to develop natural language dialogue systems. A natural language dialogue system is a system whereby humans or machines simulating humans can converse with software using either speech or text. In response to the natural language input and other inputs, a natural language dialogue system may perform an action or return information, or alter information. The purpose of this invention is to reduce the development time required to design, build or implement a natural language dialogue system. The invention can also be used to improve the performance of a prototype or existing natural language dialogue system.

5

10

15 The key feature of the invention is its ability to use examples of how the system is desired to behave and optionally semantic information.

20 The semantic information is information relating to the domain the completed application may operate in. Some examples of a domain include a voicemail application, a transaction based service such as a billpay, an information service, an auto-attendant or a directory service.

25 The semantic information may be a list of operations that the system is desired to perform along with the parameters required for the operation, and optionally the values the parameters may take. In addition the semantic information may contain information as to the phrases that indicate reference to the operation. For instance an example of limited semantic information may be add(date,time,description) to indicate an add operation that requires date, time and description parameters. The exact format of this semantic information may vary. For instance it may be in the form of a C header file, a database schema, or an interface description language. It may also be in a native format specific only to the development environment. The limited semantic information may also be a text corpora that contains the information that the natural language dialogue system may

30

35

1 reference in its operation. In this scenario WH movement may be used to transform answers into questions.

5 The invention may be capable of making assumptions about the available semantic information to infer more information,

10 although it may also use the semantic information verbatim or a combination of both.

15 In addition to semantic information related to the domain of the application, additional semantic information relating to the language to be used by the application will be available to the invention. This information may be explicit such as a lexicon or it may be implicit such as a heuristic that provides solutions that closely resemble the language in question.

20 The user of the invention may also use examples of how the system should behave, as input to the invention to enable it to develop or improve natural language applications. These transcriptions will include examples of expected user inputs to the system. The transcriptions, may or may not have sequencing information of expected phrases embedded in them, and may or may not include other actions that the system is expected to perform. Probabilities or statistics may also be included in the examples used by the invention. In addition to examples of how the system is expected to perform, the user of the invention may also provide examples of how the system should not behave.

25 Using semantic information related to the domain, plus information regarding the language the application will use, the examples of behavior or transcriptions may be preprocessed to enable the development of the application. Such preprocessing may include but is not limited to replacing portions of examples with other symbols, attaching probabilities or attaching semantic or linguistic information such as part of speech tagging or speech act information.

30

35 Once the preprocessing is performed the invention may expand on the number of examples using linguistic knowledge. Such expansion may include the addition of new phrases or the lengthening of phrases or the shortening of phrases based upon its position in the dialogue or other linguistic knowledge.

40

1 Once all of this preprocessing has occurred the invention will
then attempt to infer a grammar from both the good and bad
examples. This process is known as Grammatical Inference. The
objective of grammatical inference is to infer a grammar that is
5 compact but can generate or parse all of the good example
phrases and none of the bad phrases. In addition it is desirable to
generalize the grammar so that it includes additional phrases not
included in the examples but would be considered by the user of
the invention to be valid examples. Ideally it should do this
10 without allowing any additional phrases that the user of the
invention considers to be examples. Although this behavior is
desirable the invention may meet this requirement in the general
while still allowing a small number of bad phrases to be included
in the grammar, or not including a small number of phrases that
15 should ideally be included in the grammar.
The invention may then include the ability for the user to
examine new examples that the natural language dialogue
application can accommodate. The user may then tag these new
examples as either good or bad examples.
20 The output of the invention is software that can be run on
hardware such as computers, integrated voice response hardware
or other platforms such as voicemail systems. The output may be
in a common computing language or an abstract description
25 language. The key outputs will include grammars, slots contained
in the grammar, dialogue state machines and optionally prompts
and computer language descriptions such as C header files or any
other similar computer language or description language.
30 The speech or text input to the invention, and to applications
developed using the invention, may be in English or any other
human language, including languages not in common use, or
languages that may develop in the future such as specialist
languages designed to be used for operating or programming
machinery. The speech or text input may also include formal
35 languages, and languages that have a finite number of phrases,
and dysfunctional input, such as badly spoken written or typed
natural or formal languages. The speech or text input may relate
to one of more languages as defined above. For instance it may
be a bilingual system using English and Spanish.
40

1 In the past the problem the invention is trying to solve may be
solved by human programmers. Tools already exist that ease the
task of developers developing natural language dialogue systems.
The process of grammatical inference is also in the public
5 domain.

To assist with understanding of the invention, reference will now
be made to the accompanying drawings which show one example
of the invention.

10 Figure 1. shows one example of the development environment for
natural language dialogue systems.
Each block within figure represents a group of tasks. The arrows
connecting the blocks together show the flow of information from
15 one task to another. This information may include the semantic
information, good and bad examples of behavior, dialogue state
machines, grammars, slots and any other related information.
This flow of information and the sequence of tasks may not be as
rigid as shown in this diagram. For instance a database or file
20 system may be used to store the information and the different
processes may be invoked in any order. Some of the information
used by one process may not be used by another. One process as
shown in figure 1 may be implemented in more than one piece of
software, and one piece of software may also implement more
25 than one process as shown in figure 1.

Referring to figure 1 the application designer may begin by
30 entering semantic information K to the grammar and dialogue
generator A as shown. The grammar and dialogue generator will
then generate an initial application comprising of dialogue state
machines, grammars and slots. It may also only generate some of
these outputs and may also generate additional outputs. The input
35 to the generator may be manually entered, extracted from some
other description language or may be machine generated. The
grammar and dialogue generator may make use of an
understanding of how natural language dialogue systems behave
40 to generate these dialogues. For instance it may derive dialogue
states based on the different operations described in the semantic
information, and may describe state transitions used to extract
from the user information that is required but has not been
supplied. It may describe responses to standard phrases such as

1 "repeat", "stop", "cancel" or "operator" or natural language
2 equivalents in other languages. It may add context sensitive help
3 based upon allowable input phrases. It may accommodate
4 exception events, such as network outages between the user and
5 the end application.

10 The manual examination tool B may then optionally be used to
11 modify the generated output. In the ideal scenario manual
12 examination may not be required. Alternatively the designer may
13 chose to use the invention to modify an existing application I or a
14 hand-coded application J.

15 Example behaviors M or transcriptions can then be input to the
16 system. Using this information, combined with optional
17 additional information such as a grammar library L or semantic
18 information P such as a lexicon, the transcriptions M will be
19 preprocessed in the preprocessor C. This preprocessing will most
20 likely involve the replacement of portions of the examples with
21 non terminals representing sub-grammars. The subgrammars will
22 be derived from either the grammar library or the current
23 description of the application. The grammar library will contain
24 grammars that may be re-used between applications such as
25 dates, times and money amounts, or any other useful grammar.

25 The software description will then be passed to the symbolic
26 inference engine E. The symbolic inference engine serves the
27 purpose of extending the number of examples used by the system.
28 It will use linguistic and/or symbolic manipulation to perform
29 this. For instance it may add synonyms or antonyms extracted
30 from a lexicon.

30 An example of a feature the symbolic inference engine E may
31 perform is to accommodate co-operative answers. Co-operative
32 answers are answers to questions that don't appear to some as
33 being answers to the question asked. For instance a user may
34 respond to a question such as "What time would you like to travel
35 to Melbourne?" with an answer such as "No I wanted to go to
36 Malvern, not Melbourne". Such co-operative answers could be
37 constructed and added to the examples. Another form of co-
38 operative answer is pre-emption. For instance a user may respond
39 to the question "Do you have a fax machine I could fax the
40 information to?" with the answer "Yes my fax number is 9253

1 6788?" This type of cooperative answer could be added by
adding response from future states to the current state.

5 The grammatical inference engine F attempts to construct more
general grammar from the examples. This may be performed with
well known grammatical inference techniques such as the inside-
outside algorithm, evidence driven state merging, bayesian model
merging or hill climbing.

10 After the grammatical inference engine has constructed a
grammar, the scenario generator G may be used to construct new
examples. The manual examination tool H may be used to tag the
previously unseen examples as being either good and bad. These
new examples can then be fed back into the preprocessor C and
the process completed, until the user was satisfied with the
examples generated by the scenario generator G.

15 The application description could then be passed to the post
compiler Q. The application description R should be in a platform
independent format. The post compiler Q would then generate a
platform specific format that could be installed on the target
platform.

20 Figure 2 shows the application of the Development Environment
for Natural Language Dialogue Systems to the task of improving
an application. Referring to Figure 2 the components of the
development environment so far are listed as S. The application
25 would then be ported to the operational platform T. This platform
may be an integrated voice response unit running speech
recognition, or a computer, or a web or email server, or any other
device or apparatus that can support a natural language dialog.
The platform T may be a device that is used by the target
30 audience or a prototype platform used for alpha release software.
The number of dialogues that are supported by the application
may be less than desired at the time of the initial release of the
software. Where this is due to unexpected phrases (either text or
speech) the transcriptions of the phrases can be added to the
35 examples and the application refined. In the case of a speech
enabled application the transcription may need to be entered
manually using the analysis and transcription tool U. The
invention may or may not be integrated with other tools that aid
in the development of natural language dialogue systems.

1 The claims defining the invention are as follows:

1. A development environment for natural language dialogue systems that uses a combination of minimal semantic information of the application domain, examples of desired behavior, information about the language being used and grammatical inference to develop the application, by combining predefined grammars from grammar libraries and auto generated grammars with example behaviors by partially parsing the example behaviors before passing into a symbolic inference engine that may extend the number of examples before passing into a traditional grammatical inference engine and a scenario generator that generates new examples for manual tagging.

5 2. The development environment for natural language dialogue systems of claim 1 wherein one or more of either the symbolic inference engine, postcompiler and manual examination tool is absent.

10 3. The development environment for natural language dialogue systems of claim 1 wherein the symbolic inference engine can predict co-operative answers, through language generation and concatenating grammars from subsequent states.

15 4. The development environment for natural language dialogue systems of claim 1 wherein the partial parsing of example behaviors is performed by replacing portions of the phrase that can be parsed by sub grammars with a symbol representing that sub grammar.

20 5. The development environment for natural language dialogue systems of claim 1 wherein grammars, slots, and dialogue state machines are derived from different operations described in the semantic information, and state transitions are generated to extract from the user information that is required but has not been supplied, standard behaviors such as "repeat", "stop", "cancel" or "operator" are accommodated, and context sensitive help based upon allowable input phrases are automatically generated.

25 6. The development environment for natural language dialogue systems of claim 1 wherein similar processes are used to develop and improve applications

30 7. A development environment for natural language dialogue systems substantially as herein described with reference to the accompanying drawings.

35

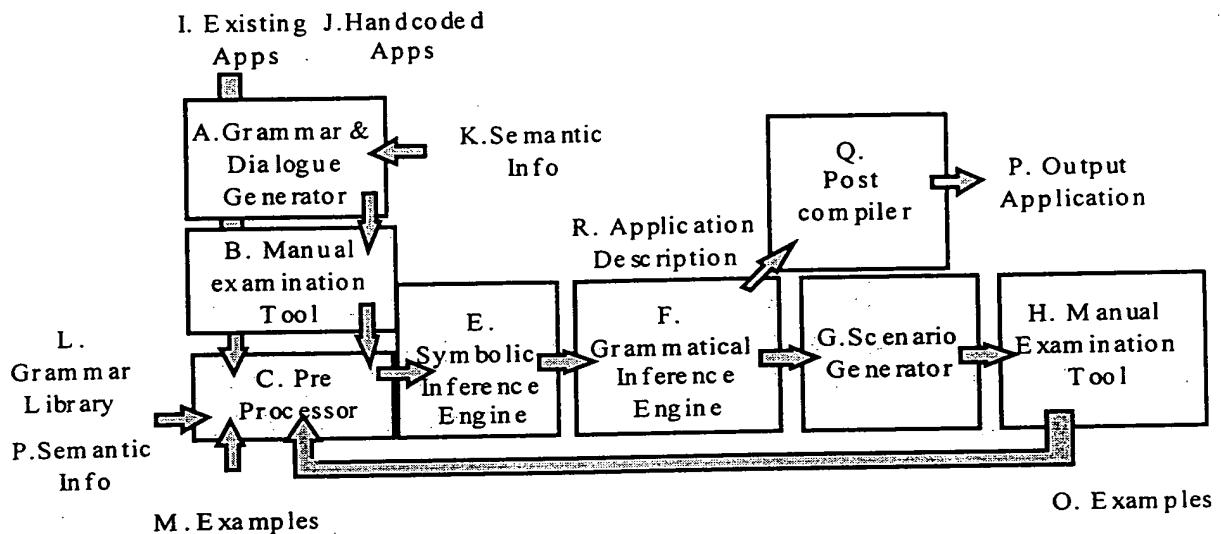


Figure 1.

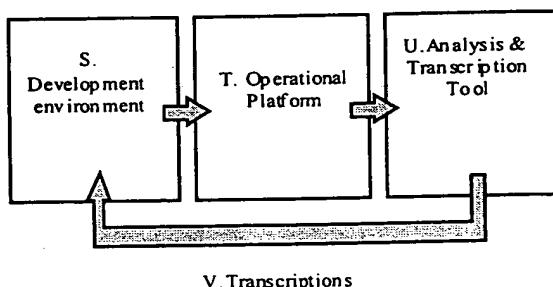


Figure 2.

ABSTRACT

A development environment for natural language dialogue systems is disclosed. The invention uses a combination of examples of desired behavior, minimal semantic information of the application domain, information about the language being used and grammatical inference to develop the application. It does this by combining predefined grammars from grammar libraries and auto generated grammars with example behaviors, by partially parsing the example behaviors before passing into a symbolic inference engine that may extend the number of examples before passing into a traditional grammatical inference engine and a scenario generator that generates new examples for manual tagging.

The symbolic inference engine may predict co-operative answers, through language generation and concatenating grammars from subsequent states.

Grammars, slots, and dialogue state machines may be derived from different operations described in the semantic information, and state transitions are generated to extract from the user information that is required but has not been supplied, standard behaviors such as "repeat", "stop", "cancel" or "operator" are accommodated, and context sensitive help based upon allowable input phrases are automatically generated. The invention uses similar processes to develop and improve applications.

THIS PAGE BLANK (USPTO)